

Motion Control: Multi-axis Functions

TM441



Prerequisites

Training modules:	TM440 – Motion Control Basic Functions
Software	Automation Studio 3.0.90 Automation Runtime 3.08 ACP10_MC Library 2.280
Hardware	None

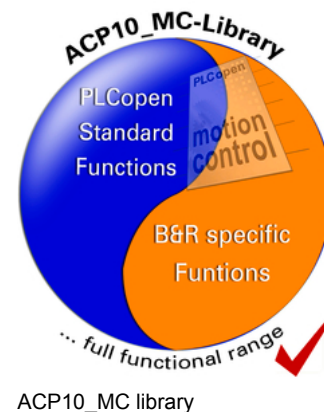
TABLE OF CONTENTS

1 INTRODUCTION.....	4
1.1 Training module objectives.....	4
2 GENERAL INFORMATION ABOUT LINKING DRIVES.....	5
3 SIMPLE LINK.....	7
3.1 Linking with reference to the start position.....	9
3.2 Dynamic phase shift.....	12
4 ELECTRONIC CAM PROFILES.....	15
4.1 Introduction.....	15
4.2 Creating cam profiles.....	16
4.3 Linking functions.....	19
4.4 Predefined sequences.....	23
5 CAM PROFILE AUTOMAT.....	25
5.1 Introduction.....	25
5.2 Structure and functionality.....	27
5.3 Implementing the Cam Profile Automat.....	29
5.4 Compensation gear.....	37
5.5 Examples.....	38
6 SUMMARY.....	39

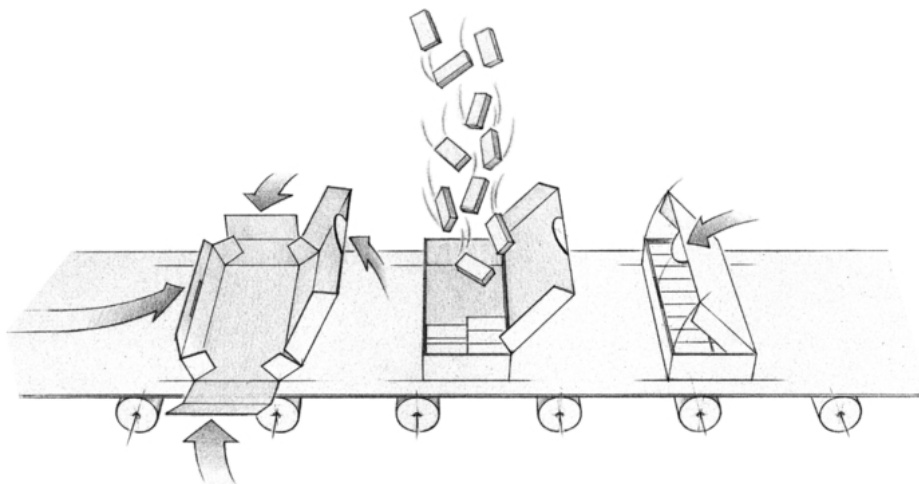
1 INTRODUCTION

The B&R drive solution provides flexible, high-performance tools for linking drives electronically. This makes it possible, for example, to create synchronous drives that are linked together for linear as well as for dynamic movements. There are many practical applications for these solutions, such as synchronous cutting procedures, dynamic transfer processes and flexible length partitioning.

As usual, the ACP10_MC library provides a comprehensive selection of function blocks that can be used to operate these functions.



This training module explains how to use a range of functions to configure and control electronically linked movement sequences.



Cartoning

We will begin with a brief overview to become familiar with the individual options. We will then cover some basic theory, ask ourselves some important questions, and finally, learn how to use multi-axis functions in actual applications.

1.1 Training module objectives

This training module will use descriptions and examples to explain the use of ACP10_MC multi-axis functions.

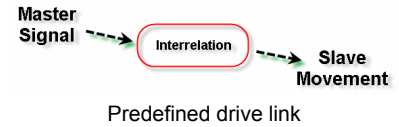
You will learn ...

- ... possible applications of motion control multi-axis functions.
- ... how to link drives and implement specific linking operations.
- ... the procedure for creating linear and non-linear cam profiles.

2 GENERAL INFORMATION ABOUT LINKING DRIVES

Linking drives electronically results in a predefined synchronized movement.

Drive A is linked to drive B via position. This means that while the drives are actively linked, drive A must adjust its position in a specified manner according to the position of drive B. In this case, drive B is the master, which specifies a reference position, and drive A the slave, which has a position based on the master position.



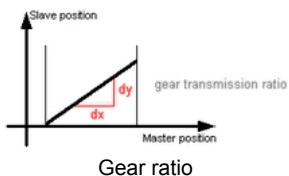
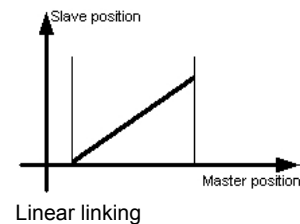
A drive link requires a master signal, which provides the reference (position, time) and at least one slave drive, which must follow this reference value using a specific "rule". However the master signal does not have to come from an actual drive as it does in this example. In principle, drives can also be linked to a variety of suitable reference values.



The master remains unaffected by the linking procedure. It is simply used as the basis for the desired linking signal. For example, if a drive's position value is used as the master signal, then this master axis can still be given a command even while the drive link is active. In this situation, the slave drive is still completely dependent on the master signal.

The shape of the position link (i.e. the "position rule" that tells the slave drive how it must follow the master signal) can be displayed in a diagram with a comparison of the master and slave position.

This is shown in the image for a linear relationship between the master and slave position:

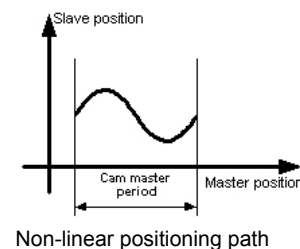


The position of the linked master is shown in the horizontal direction. The position of the linked slave can be seen in the vertical direction.

According to this link, when the master signal changes uniformly (e.g. the master axis moves at a constant speed), the speed of the slave axis is also constant.

In this case, we are talking about an "electronic gear", a type of link which is used quite often. The gear ratio is represented by the slope of the "linear curve".

However, the position relationship does not have to be linear. In principle, electronic cam profiles can be created and used for any positioning paths necessary.



General information about linking drives

A simple link for an electronic gear as well as a link via cam profiles can be quickly implemented for the drive. Automation Studio provides a cam profile editor for creating user-specific cam profiles. The corresponding function blocks used to configure and control the drive links can be found in the ACP10_MC PLCopen library.

The Cam Profile Automat offers extensive settings for connecting multiple cam profiles to each other.



The multi-axis functions in the PLCopen library are operated just like the function blocks that we are already familiar with. Implementation of these functions in the automatic sequence of an application program is therefore also the same.



[Programming \ Libraries \ Motion libraries \ ACP10_MC \ Function Blocks \ Overview of the Supported Function Blocks](#)

3 SIMPLE LINK

The functions in the ACP10_MC library for controlling electronic gears are quite easy to use.

The functions for linking axis objects require the axis ID for master and slave. As usual, the NC objects defined in the NC mapping table (real or virtual axis) can be accessed using the address of the global variables with the same names.

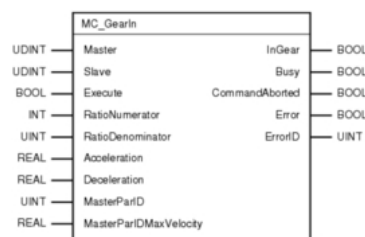
NC Object Name	Network Interface	Node Number	Advanced	NC Object Type
Axis1	SS1.IF2 [Powerlink]	1		ncA_XS
Axis2	SS1.IF2 [Powerlink]	2		ncA_XS
Axis3	SS1.IF2 [Powerlink]	1		ncV_A_XS

NC objects in the NC mapping table

To prepare the axes for movements, they normally need to have power applied (controller switched on) and have a homing procedure carried out.

MC_GearIn function block

This function block is used to start a linear link.



MC_GearIn function block

Input	Description
Master	Specifies the master axis reference.
Slave	Specifies the slave axis reference.
Execute	Start link with positive edge on the Execute input.
RatioNumerator RatioDenominator	Gear ratio of the link. For example 3/1 → Slave moves 3 times faster than the master.
Acceleration Deceleration	Slave limit values when linking and changing the gear ratio.
MasterParID	A ParID can be used as master reference instead of the master set position.
MasterParIDMaxVelocity	When using a MasterParID, this parameter specifies the maximum speed of this ParID value, which applies during the transition into the gear and when changing the gear ratio.

Table: Overview of MC_GearIn input parameters



The state of the master axis is not affected by the link.

Link parameters cannot be changed if the master moves backwards in the active link.

The link cannot be started if the master is moving backwards!

Link solutions

There are different solutions for the link depending on the application.

If the slave axis should continue running at the current speed, then **MC_GearOut** can be called.

If the slave axis should be stopped, then **MC_Halt** or **MC_Stop** can be called.

If the slave axis should execute a basic movement, then one of the **MC_MoveXYZ** functions can be called.



The state of the slave changes to "Synchronized Motion" when the link is started successfully. When the link is terminated using **MC_GearOut** the drive maintains its current speed and the state changes to "Continuous Motion".

Therefore, the **MC_Stop** function block would also have to be used to stop movement of the slave axis.



[Automation Software \ Getting Started \ Create a motion application in Automation Studio \ Coupling two axes Achsen](#)

[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Electronic gears](#)

Exercise: Simple axis link

Create a simple axis link in which the slave axis follows the master axis during active linking at a 1:1 ratio. Use the library examples for the master and slave axis.

- 1) Add the master axis
- 2) Add the slave axis
- 3) Add the library example for a single axis (master)
- 4) Add the library example for a simple link between two axes (slave)
- 5) Transfer the program.
- 6) Switch on the controller for both axes
- 7) Complete the homing procedure for both axes
- 8) Start a positive movement on the master
- 9) Link the slave axis and monitor the slave position
- 10) Stop the master axis
- 11) Start an additive movement of 1000 units
- 12) Check the slave position

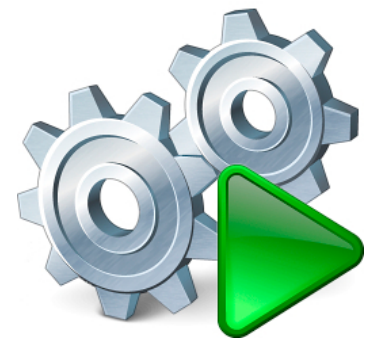


The master set position is used by default. Depending on the application, other master signals (e.g. actual position of an encoder, etc.) can also be specified on the FB input **MasterParID**.

3.1 Linking with reference to the start position

The MC_GearInPos function block is an extension of the MC_GearIn function block. Some applications require a defined position for the start of the drive link.

MC_GearInPos is used to define both the master and the slave position for the start of the electronic gear. This makes it possible to achieve a defined "position" of the axes relative to each other for the start of the drive link.

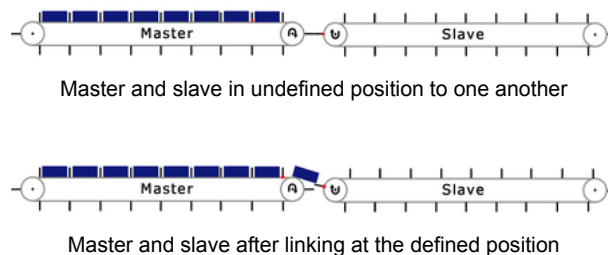


Drive link with position reference

This functionality can be used for the following example:

A conveyor belt with product receptors, with a distance of one period between receptors, accelerates from a standstill to match the speed of the preceding belt from which it receives the products.

In this case, the product must always be transferred at a defined position. When starting the process, MC_GearInPos makes sure that the slave position and speed is in the correct reference to the master at the defined master position.



Values for the position period and factor can be entered in the NC mapping table for `PLCopen_ModPos="<Period>,<Factor>` to adjust the position value:

NC INIT Parameter	ACOPDS Parameter	Additional Data
ax1_init	ax1_par	<code>PLCopen_ModPos="<Period>,<Factor>"</code>

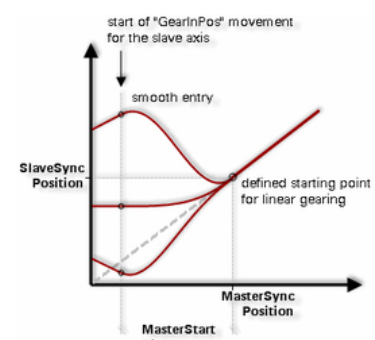
Advanced setting for position period and factor in the NC mapping table

MC_GearInPos function block

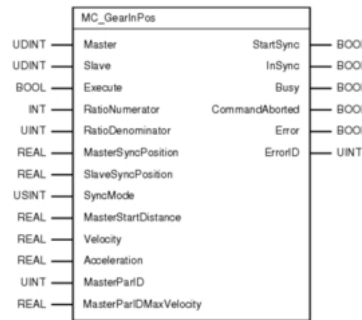
The image shows the procedure for various initial situations resulting from a random position of the slave axis.

The entry movement of the slave drive is started when `MasterSyncPosition - MasterStartDistance` has been reached.

At this point there is a smooth transition between the entry movement and the respective gear ratio.



Smooth transition into linked movement



MC_GearInPos function block

Input	Description
Master Slave	Specifies the reference for the master/slave axis.
Execute	Start link with positive edge on the Execute input.
RatioNumerator RatioDenominator	Gear ratio of the link. For example 3/1 → Slave moves 3 times faster than the master.
MasterSyncPosition SlaveSyncPosition	Position at which synchronized master and slave movement begins
SyncMode	Defines the type of synchronization.
MasterStartDistance	The distance within which the system has to perform a "smooth" transition into the gear link ("compensation movement of the slave").
Velocity Acceleration	Maximum speed or acceleration for the slave when entering the link.
MasterParID	A ParID can be used as master signal instead of the master set position.
MasterParIDMaxVelocity	When using a MasterParID, this parameter specifies the maximum speed of this ParID value, which applies during the transition into the gear and when changing the gear ratio.

Table: Input parameter MC_GearInPos



An active link made by the MC_GearInPos function block cannot be interrupted by an additional function call for the same or another instance and the gear ratio also cannot be changed.

The slave must be at a standstill when starting the link with MC_GearInPos!

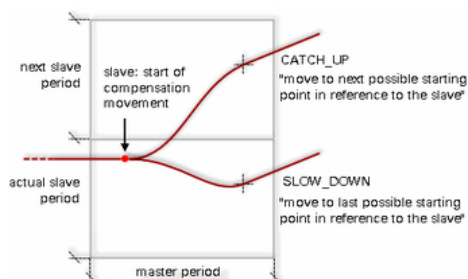
The link cannot be started if the master is moving backwards!

The master axis is not affected at all by these actions and can therefore execute basic movements as usual.

Synchronization modes

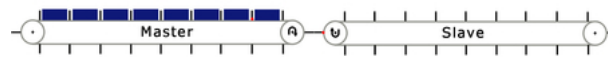
A mode parameter can also be used to define the position period in which the slave drive should transition into the gear. The mode (current, last or next period) determines the actual point where the drives are linked, depending on the current position of the slave axis within the position period. This allows e.g. a slave axis compensation movement to the corresponding connection point to be performed one period before or one period after the current position period:

As shown in the image above, CATCH_UP always initiates a movement to the next drive linking point. SLOW_DOWN always initiates a movement to the preceding drive linking point. Depending on the current position of the slave, the CATCH_UP mode might make it necessary to change to the next period (see above). It is also possible to change to the preceding period for the SLOW_DOWN mode.



CATCH_UP and SLOW_DOWN diagram

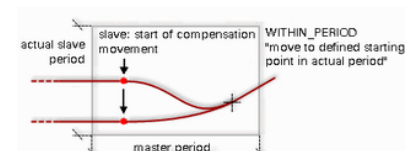
Based on our conveyor belt example, the slave would move forward when linking in CATCH_UP mode and first backwards then forward in SLOW_DOWN mode.



Conveyor belt example

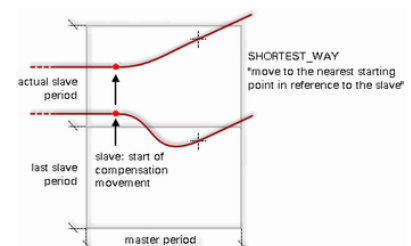
The image shows the behavior in WITHIN_PERIOD mode for two different starting situations. In these cases, the slave always moves to the starting point within the current period.

The slave would always move backwards when linking in WITHIN_PERIOD mode because the SlaveSyncPosition is at the beginning of the period in our conveyor belt example.



WITHIN_PERIOD diagram

When in SHORTEST_WAY mode, the slave always moves to the next closest drive linking point. This is illustrated in the image for two different starting situations. Depending on the situation, it could be necessary to change the preceding or the next period.



SHORTEST_WAY diagram

What happens when the master has already passed the starting position for compensation? When using periodic axes, this starting point also returns to the next period. Otherwise, the function block sends a corresponding response (Error).



When using periodic axes, the SyncMode input must always be configured to prevent an error from being generated. If a non-periodic axis is used, then the SyncMode input parameter is simply ignored.



Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Electronic gears \ MC_GearInPos

Exercise: Electronic gear with reference to the start position

The MC_GearInPos function block can now be tested the same way as in the previous example.

Integrate this function block and operate it using the variable monitor. Start the function block while the slave axis is idle. Observe how the slave behaves.

3.2 Dynamic phase shift

The **MC_Phasing** function block creates a phase shift in the master position of the slave axis if a link is active. The master position is shifted with respect to its actual physical position.

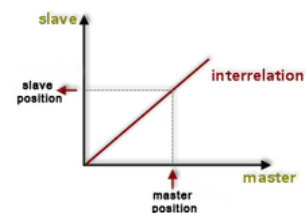
The phase shift can only be "seen" by the slave, the master doesn't notice. The phase shift remains in place until another phasing command changes it.

MC_Phasing can be used if a link has already been started with **MC_GearIn**, **MC_GearInPos** or **MC_CamIn**.

How is this done?

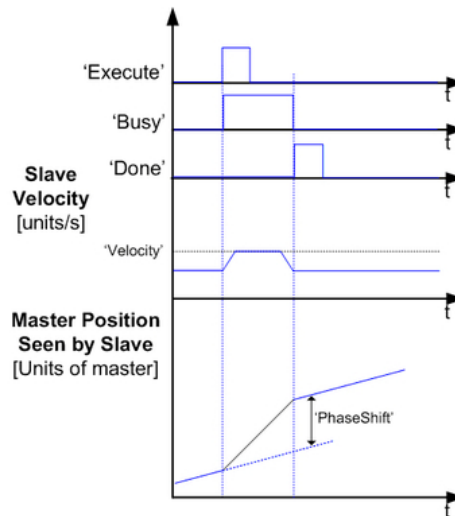
The slave position is determined by the position of the linked master and the link ratio (linear or via cam profile).

The MC_Phasing function block generates a value for an additive element or additive master axis. This element is added to the actual position of the master. The resulting value is then applied to the master side of the link ratio.



Master - Link ratio - Slave

When the function block is activated, the specified target value for the phase shift is approached using a constant movement profile. The movement profile position is continually added to the actual master position, which prevents jumps in position on the slave axis. The master axis is not affected by this action at all.



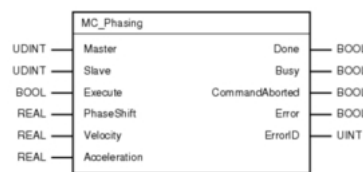
Targeted phase shift generated by the MC_Phasing function block

This changes the set position for the linked slave. A targeted phase shift can therefore be implemented. This smooths out the additive master axis value that is generated to achieve the desired end value.



MC_Phasing can be used to sort products. After sheets of cardboard have been cut, they lie end-to-end on a conveyor belt. They are then transferred to a second conveyor belt. As each sheet reaches the second belt, a phase shift can be implemented. This creates a gap between the products, which is required for further processing.

MC_Phasing function block



MC_Phasing function block

Input	Description
Master Slave	Specifies the reference for the master/slave axis.
Execute	Phase shift is started at a rising edge.
PhaseShift	Phase shift [master's units]
Velocity Acceleration	Maximum speed / acceleration for achieving the phase shift [units/sec].

Table: MC_Phasing entry overview



The resulting slave position is directly dependent on the link ratio. For example, the gear ratio for an electronic gear has the following effect on the result:

Gear ratio = 1:5 (Master:Slave)

Master-side shift = 2000 units (additive master axis)

Shift on the slave side = 10,000 units

Similar function blocks:

- MC_BR_Phasing
- MC_BR_Offset



[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Phase shift and offset shift](#)

Exercise: Phase shift

Use the MC_Phasing function block. Go through all of the steps up to the point of activating the link and perform the phase shift for different settings.

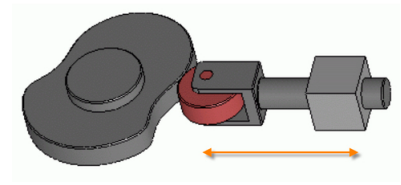


The phase shift is additive to the current movement. The "MC_Phasing" function block can also be tested while the master axis is idle.

4 ELECTRONIC CAM PROFILES

To implement dynamic, non-linear movements, the B&R drive solution offers the option of using electronic cam profiles for axis linking. These cam profiles can be created by the user.

Electronic cam profiles can be used many different ways.



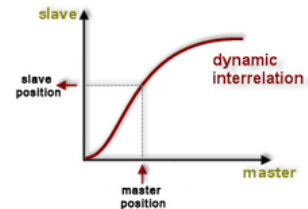
Mechanical cam (Silberwolf / de.wikipedia.org)



Cam profiles can be used quite effectively for spring winding machines. Separate axes are used to control the feed, curvature and slope respectively. This makes it possible to create any shape needed (slopes, cones, etc.)

4.1 Introduction

In the cam diagram, we see the master position value in the horizontal direction and the slave position in the vertical direction. The cam profile now assigns a respective slave position value for each master position value within a defined range (cam profile master period / cam master period). The slave drive must follow this profile while the drives are actively linked.



Cam profile as position relationship

The master position is converted to a corresponding slave position via the cam profile. This allows the master to move in both directions. The slave drive is "tied" to the master via the cam profile.

This means that the speed and acceleration values for the slave drive are also based on the speed and acceleration of the master in connection with the cam profile.



Therefore, the entire course of the cam profile must be checked to make sure the slave drive can handle any speed and acceleration values that might occur.



Maximum speed that occurs on the slave

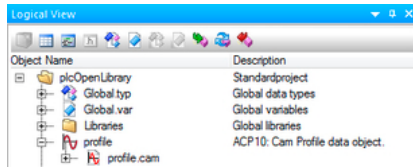
Let's assume the master signal changes at a constant rate: That means, for example, a uniform movement or time for a master axis.

Critical ranges (with maximum values for slave speed or acceleration) are represented in the cam profile by maximum slope (-> speed as first derivative of the position) and the maximum slope change (-> acceleration/deceleration as second derivative of the position) due to the position comparison.

4.2 Creating cam profiles

Automation Studio has a powerful cam profile editor for creating cam profiles. Cam profiles can be edited in the cam profile editor as soon as they have been inserted to the project.

Cam profiles are created in Automation Studio as NC software objects and transferred to the controller where they can be selected during runtime.



Cam profile objects in the logical view



It is generally advisable to create standardized cam profiles. These have end points with a ratio of 1:1 or 1:0. This allows the cam profile to be stretched in unit scaling as needed using corresponding function blocks or the Cam Profile Automat. As a result, they can be used on a wide range of axes with different scaling.



[Motion \ Project creation \ Motion control \ Connection type \ Cam profiles \ Cam profile editor](#)

4.2.1 Editing a cam profile

The cam profile editor in Automation Studio is a full-featured tool that provides a clearly arranged and precise environment for creating cam profiles to link axes according to very specific requirements.

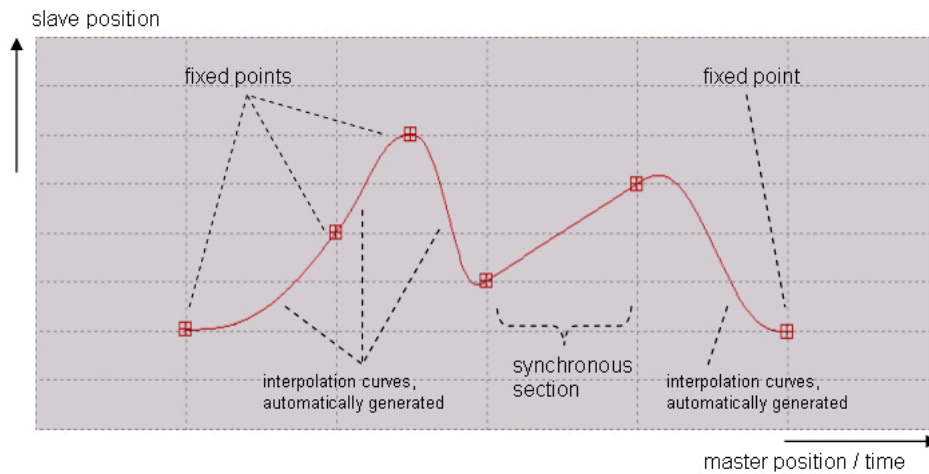
The following properties can be configured in the cam profile editor:

- General properties
- Color settings
- Extension
- Display options
- Labels and formulas
- Characteristic values for curves
- Notations in the diagram

The following functions are provided in the cam profile editor:

- Fixed points
- Synchronous sections
- Interpolation curves
- Importing mechanical cam profiles

It is now possible to define fixed points on the curve as well as synchronous sections (linear sections along the path of the curve) to create a cam profile. The cam profile editor automatically integrates these into an overall cam profile using interpolation curves.



Structure of a cam profile

The image shows an example. A total of four fixed points and one synchronous section have been defined. The cam profile editor automatically integrates these definitions into a complete cam profile. It does this by calculating and displaying interpolation curves. The user can even specify the form of the interpolation curves.



[Motion \ Configuration \ Motion Control \ Connection type \ Cam profiles](#)

[Motion \ Configuration \ Motion Control \ Connection type \ Cam profiles \ Cam profile editor](#)

Fixed points

A fixed point is a point in the cam profile for which the user defines the desired position of the slave axis in relation to a specific position of the master axis.



The notation indicates whether position or time units should be used in the diagrams on the horizontal axis (i.e. the master axis). The use of position is referred to as "mathematical notation". The use of time is referred to as "physical notation" (comparable to a constant master speed).

Therefore, in physical notation, the first derivative in the fixed point is equal to the speed and the second derivative in the fixed point is equal to the acceleration of the slave axis. The cam profile represents the path-time diagram of the slave axis.



[Motion \ Project creation \ Motion control \ Connection type \ Cam profiles \ Cam profile editor \ Fixed points](#)

Synchronous sections

A synchronous section is a section in the cam profile where the user specifies a linear path for the master and slave positions.

A constant master axis speed within a synchronous section also results in a constant slave movement. In other words, the cam profile is linear (comparable to an electronic gear).



When using physical notation (master axis = time) the master position is entered as a time value. The slope of the synchronous section corresponds to the speed of the slave axis in this section. (→ time passes "evenly")



Motion \ Project creation \ Motion control \ Connection type \ Cam profiles \ Cam profile editor \ Synchronous sections

Interpolation curves

The section of a cam profile calculated by the cam profile editor to connect two defined elements (fixed points, synchronous sections) is called an interpolation curve.

After each new fixed point or synchronous section is entered, an interpolation curve integrating it in the rest of the profile is automatically calculated and displayed. The cam profile editor makes sure that there is exactly one interpolation curve between any two defined components.

Likewise, when a fixed point or a synchronous section is deleted, any extra interpolation curves are also deleted.



The calculation ensures that the cam profile function and its first derivative are constant at the transition points (e.g. the curves do not contain any jumps at the end points).

Various curve types can be selected for each interpolation curve to fine-tune the profile between the defined areas (fixed points and synchronous sections). These provide different predefined shapes according to the type. Specific curve profiles can be implemented using type-specific settings (turning points, joining points, etc.).



Motion \ Project creation \ Motion control \ Connection type \ Cam profiles \ Cam profile editor \ Interpolation curves

Importing mechanical cam profiles

It is often necessary to replace mechanical cam profiles with electronic ones or to reproduce content from a CAD system electronically. The cam profile editor makes it possible to import and then export interpolation points, which allows calculated curves to be reused in CAD, for example.

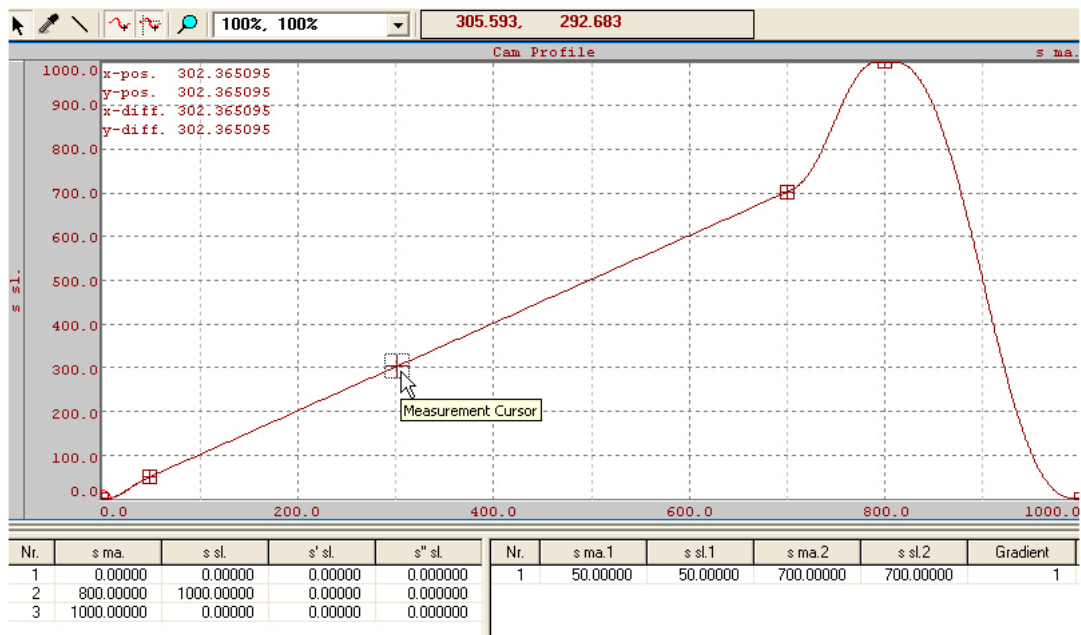


Motion \ Project creation \ Motion control \ Connection type \ Cam profiles \ Cam profile editor \ Mechanical cam profiles

Exercise: Create a cam profile

Insert a cam profile into your project and edit it in the cam profile editor. Do this using the option for defining fixed points and synchronous sections.

The image shows a movement profile in which the slave axis reaches its maximum position in the right quarter of the profile.



Example of a cam profile

When creating your cam profile, make sure that the profile has the same slope at the start and end points. This characteristic is important for the following drive link applications.



Transfer the project containing the new cam profile to the controller and activate monitor mode. The cam profile should now appear in the project software configuration as a data object on the controller.

The cam profiles needed for the link application must be transferred to the drive using the **MC_CamTableSelect** function block.

4.3 Linking functions

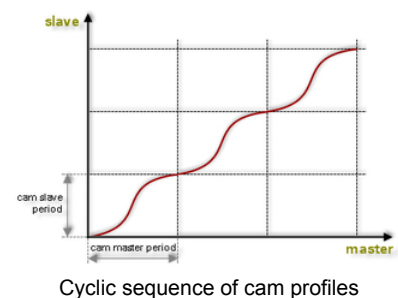
A cam profile must be transferred from the controller to the slave drive before it can be used.

The following section will explain the corresponding routines and procedures needed to do this.

4.3.1 Preparing cam profiles

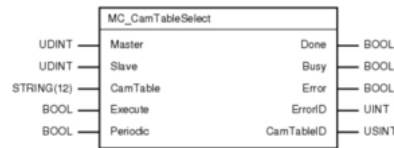
The **MC_CamTableSelect** function block is required for transferring a cam profile object to the linked slave. When this function block is called, the corresponding cam profile is transferred and an ID is returned for further use with the link function.

This function block is also used to set whether the cam profile should be processed one time or cyclically. The cam profile can be set to repeat itself cyclically. This results in a continuous positioning loop for the slave.



MC_CamTableSelect function block

This function block is used to download and configure a cam profile on the linked slave.



MC_CamTableSelect function block

Input	Description
Master	Specifies the master axis reference.
Slave	Specifies the slave axis reference.
CamTable	Name of the desired cam profile.
Execute	Activate function block with positive edge on the Execute input.
Periodic	Select one-time or cyclic processing of the cam profile. <ul style="list-style-type: none"> • mcNON_PERIODIC • mcPERIODIC

Table: Input parameters for MC_CamTableSelect

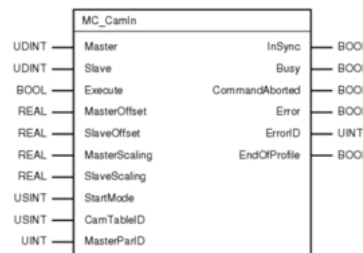


Smooth entry must be guaranteed when starting a cam profile link. When a sequence of cam profiles is created, the end of one cam profile leads directly into the beginning of the next. It is therefore important to ensure that the speed and acceleration of the transition is consistent. Sharp angles must not be allowed to occur anywhere in the positioning path.

The following sections will provide some examples of how to do this.

4.3.2 Linking cam profiles

A cam profile on the ACOPOS is linked using the **MC_CamIn** function block.



MC_CamIn function block

Input	Description
Master	Specifies the master axis reference.
Slave	Specifies the slave axis reference.
Execute	Start link with positive edge on the Execute input.
MasterOffset	Offset on the master side
SlaveOffset	Offset on the slave side

Table: Input parameter MC_CamIn

Input	Description
MasterScaling SlaveScaling	Master / slave-side scaling of the cam profile.
StartMode	Start mode based on the offset
CamTableID	Cam profile ID of the desired cam profile. This is provided by the MC_CamTableSelect function block once the cam profile has been successfully downloaded.
MasterParID	A ParID can be used as the master signal instead of the master set position.

Table: Input parameter MC_CamIn

Ways to activate the link

Similar to the MC_GearInPos function block, the exact starting point for the link can also be defined here in relation to the master and slave position. The two parameters Offset and StartMode can be defined. An overview of the modes has been provided. A detailed description is included in the Automation Studio Online Help documentation.

Mode	Description
mcABSOLUTE	Absolute from the zero point of the position period "Zero point of the position period + Offset"
mcRELATIVE	Relative to the current position "Current axis position + Offset"
mcDIRECT	Directly from the current master / slave position

Table: Overview of start modes



When using mcDIRECT start mode, the master axis must be at a standstill to link the slave at the correct position!



Since the slave requires a certain amount of time to reach its starting position, the master may pass its starting position several times before it is ready. Once the slave has reached its starting position, the link is started as soon as the next master starting position has been reached.

As a result, activation of the link may be shifted by a number of master periods depending on the situation. The master remains completely unaffected.



[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cams \ MC_CamIn](#)

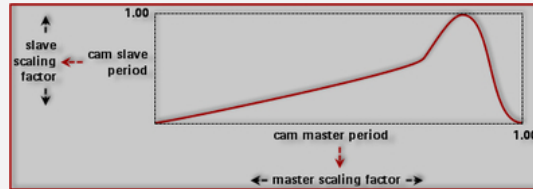
Scaling the cam profile

A cam profile can be "stretched" on both the master and slave side in order to create the link. The input parameters for the gauge factors are included in the MC_CamIn function block.

This extends the length of the master and slave cam profile by the corresponding factor.



Cam profiles are often created with a master-side extension of one unit (cam profile master period = 1) and a slave-side extension of one unit (cam profile slave period = 1). This makes it rather easy to "stretch" a cam profile to match a particular process:



Scaling the cam profile using the gauge factor

For example, let's assume that the cam profile (master period=1, slave period=1) should be set in a way so that exactly one cut is made for each master axis revolution. Therefore, the gauge factor for the master-side must be set equal to the number of units for a master revolution.

When using a linear cam profile, the "gear ratio" can be determined using the gauge factors.

The MC_CamOut function block can be used to terminate an active link. Additionally, linking can be deactivated by stopping the slave axis or executing a basic movement on the slave axis.

Exercise: Axis linking via a cam profile

Using the method described, set up the link for the cam profile you created earlier. The cam profile should be processed cyclically.

- 1) Add the basic movement example for the master axis
- 2) Add the cam profile example for the slave axis
- 3) Start a positive movement on the master axis
- 4) Trace the position on the slave axis
- 5) Replace the cam profile with the one created earlier
- 6) Trace the position on the slave axis



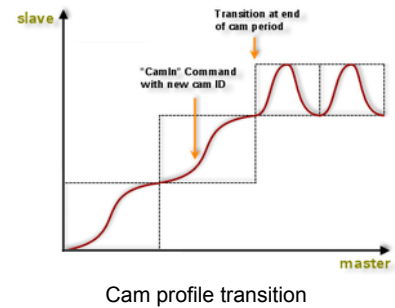
Programming \ Examples \ Motion \ Motion Control \ Linking two axes with a cam profile

4.3.3 Changing cam profiles

When a cam profile link is active, the cam profile can be changed by calling the MC_CamIn function block again.

When a new CamTableID is entered and a positive edge occurs on the Execute input, the period of the active cam profile is ended and the new cam profile begins. The end point of the first cam profile is the start point of the second cam profile.

Neither the master and slave offset nor the start mode have any effect on the cam profile change.



Here too, it is important to ensure a smooth transition between cam profiles in order to avoid jolts.

Once the required cam profiles have been transferred to the respective drive (once or cyclically), they can be arranged in any sequence needed. The routines for changing the cam profile must be performed in the application program at the corresponding times.

Exercise: Cam profile change

We can now test this function without having to complete any additional preparation. Link the cam profile as in the previous example.

Execute the MC_CamIn function again, but now change the values for the cam profile scaling. In principle, this procedure is carried out the same way as linking a new cam profile.

4.4 Predefined sequences

Applications can be created more easily using predefined processes and the predefined cam profiles on the drive.

Function block	Description
MC_BR_CamDwell	Cam profile link between master and slave axes, dwell times can be defined, configurable entry and exit movements, definable compensation distances
MC_BR_AutoCamDwell	Like MC_BR_CamDwell, a cam profile does not have to be specified, the slave drive calculates a jolt-minimized movement profile
MC_BR_CamTransition	Cyclically repeating cam profile compensation process with optional entry and exit transition
MC_BR_CrossCutterControl	Control and correct cutting axes, combination of phasing and offset possible

Table: Overview of predefined processes



Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cams \ Predefined sequences

5 CAM PROFILE AUTOMAT

The Cam Profile Automat allows event controlled linking of electronic cam profiles.



A Cam Profile Automat is already used in the background for gear links or MC_CamIn as well as the previously described predefined processes (4.4 "Predefined sequences"). The automat configuration used is described in detail in the "Additional information" section of the function block description in the Automation Studio Online Help.

The following example moves step-by-step through a more in-depth explanation of Cam Profile Automat functions.

5.1 Introduction

Steps to follow when using MC_CamIn

First, let's look at a procedure using a packaging machine.

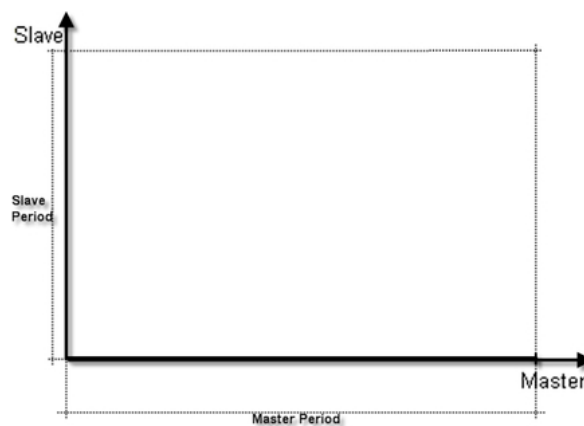
The product transporter acts as the master axis. The slave axis closes each plastic container with a cap. A high-speed digital input (trigger) detects if a product is present. If no product is present, then the slave remains in standstill. Otherwise the container is capped with a cap.



In the following section, we'll think about how we could solve this example with the MC_CamIn function block

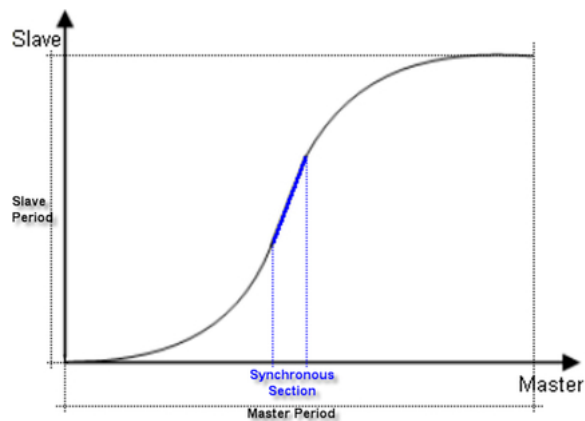
First, we need two cam profiles:

- Cam profile 1, which keeps the slave at a standstill when a container is not present.



First cam profile for keeping the slave at a standstill

- Cam profile 2 for the application of the cap:



Second cam profile for applying the cap

First, the two cam profiles must be transferred to the ACOPOS. Then a control program must be used to check if a trigger signal has been received:

- If so, cam profile 2 must be linked using the MC_CamIn function block.
- If there is no trigger signal, then cam profile 1 should be linked via the MC_CamIn function block (by changing the cam profile ID on the CamTableID input).

Efficient solution

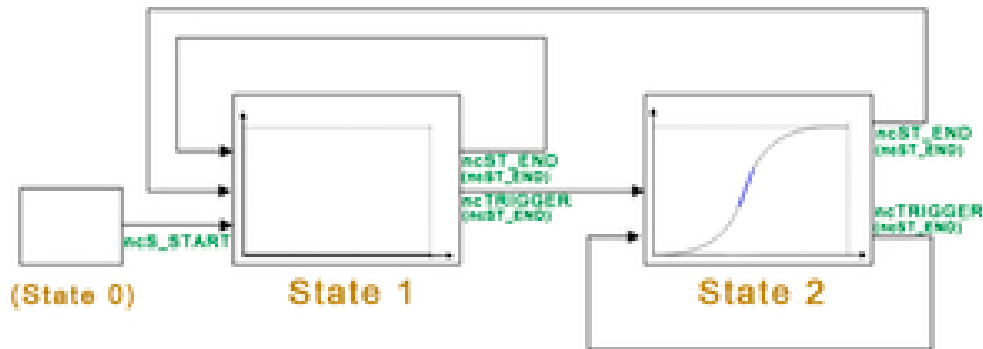
A much simpler and more efficient method would be if the drive could decide on its own which cam profile should be processed based on the current process situation. This would simplify the control program and enable much faster reaction times.

It is precisely situations such as this for which the Cam Profile Automat was created. It is initialized and configured on the corresponding slave drive, where it can then be processed independently. This keeps the CPU load comparably low, even when a large number of axes are in use. The running process benefits from minimal reaction times. There also many ways to intervene in the processing of active automats.

In the following section, we will take a step-by-step look at the structure and operation of a Cam Profile Automat using the example shown earlier.

5.2 Structure and functionality

The example can be structured in the Cam Profile Automat as follows.



Cam Profile Automat structure for the bottle capping machine

Automat states

The two cam profiles are now each packed in a specific state. These are called Cam Profile Automat states. This results in **State1**, in which the slave should not execute movements because the product is not present and **State2**, in which the bottle capping process should be executed. **State0** is optional and can be used as a starting state or a waiting state. This state does not have a cam profile assigned to it.

In terms of the master, the length of a state corresponds to the time it takes to transport a product, i.e. waiting for the next product. In terms of the slave, the length of a state corresponds to the distance required to complete the capping process.

Change events

A change event is a defined event that should cause a change of state (e.g trigger event ncTRIGGER, or reaching the end of the state ncST_END, etc).

The user must also define when the change should take effect. For example, it can take place at the end of the state (ncST_END) or immediately (ncAT_ONCE) when the event occurs. The new state that should be changed to must also be defined. The end result is an entire series of automat states. Two change events have been defined for each of the two states in our example.

Preset cam profiles

The preset cam profiles are already available on the drive and do not have to be transferred there.

CamProfileIndex	Description
0xFFFF	This preset linear cam profile can be used with a master and slave length of one unit as the CamProfileIndex when configuring the automat. This can be used with gauge factors to produce any m:n straight line.
0xFFFE	This predefined point cam profile can be used as the CamProfileIndex when configuring the automat. This point cam profile can only be used when compensation mode is enabled. It cannot be used in states with CompMode = ncOFF. The master and slave interval length of this preset cam profile is zero. However, the slope of the curve is not zero, but can be set using the gauge factors. This allows you to create applications that only require one compensation procedure without a cam profile.

Table: Overview of preset cam profiles

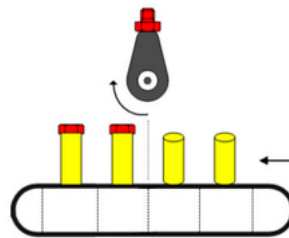
Cam Profile Automat sequence for the bottle capping machine

The bottle capping machine changes to State1 after the event-controlled start (start at a specific master position) in State0.

The slave does not perform any movements in State1. The first bottle must therefore be left out when starting the machine. If a trigger signal (ncTRIGGER) is detected during processing of State1, then the machine changes to State2 at the end of State1 (ncST_END), at which point the capping process is then executed. During execution of State2, one bottle is capped. If another trigger signal is detected during this state it is repeated.

If a product is not present or if State2 runs completely to the end (ncST_END) without a trigger signal having occurred, then the machine must switch to State1 so that the slave does not perform a movement. The automat remains in State1 until a new trigger signal is received. When a new trigger signal is received, the automat is switched back to State2 and the capping process is continued.

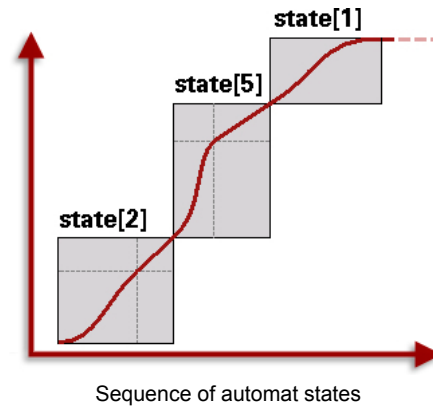
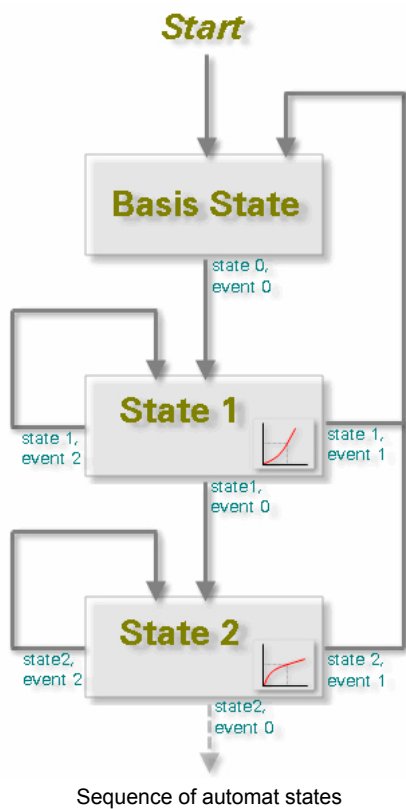
It is also necessary to ensure that the signal arrives on time within a product interval. If multiple signals are sent, this is only evaluated as one signal at the end of the state.




Bottle capping machine

This makes it possible to consecutively arrange a wide variety of cam profiles in a manner similar to the steps (states) of a step sequencer. This further enables the implementation of flexible machine processes.

Once the automat parameters have been set, the automat can then be started in any state and runs through the individual states according to the defined change events and subsequent states.



 [Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \](#)

5.3 Implementing the Cam Profile Automat

The parameters for the Cam Profile Automat can be set in two ways:

- Using the function blocks in the ACP10_MC library.
- Using the MC_AUTDATA_TYP data structure.

The data structure contains all of the Cam Profile Automat parameters in structured form. A variable with this type can be created in the application program and used to configure the automat. Steps for implementation are listed in the following table.

Procedure	Implementation
Define the global parameters for the automat Master and slave axis, start state, ...	MC_AUTDATA_TYP>.<parameter> or MC_BR_InitAutPar
Download all of the cam profiles to the drive that are used in the automat.	MC_BR_DownloadCamProfObj
Definition of the automat states	<MC_AUTDATA_TYP>.State[x] or MC_BR_InitAutState
Definition of the desired change event for each state	<MC_AUTDATA_TYP>.State[x].Event[y]

Table: Implementation procedure

Procedure	Implementation
	or MC_BR_InitAutEvent
After the above steps have been performed, the automat can then be started and operated.	MC_BR_AutControl MC_BR_AutCommand

Table: Implementation procedure



An example of a Cam Profile Automat comes with Automation Studio. It is based on the MC_AUTDATA_TYP data structure and function blocks listed here. It can be configured to handle a number of different applications.



[Programming \ Examples \ Motion \ Motion control \ Cam Profile Automat](#)

[Programming \ Examples \ Motion \ Motion control \ Cam Profile Automat \ Automat configurations](#)

- [Labeling machine](#)
- [Flying saws](#)
- [Cutting unit](#)

5.3.1 Defining global parameters

The global parameters are settings that apply to all automat states. The global settings for the automat are made using either the **MC_BR_InitAutPar** function block or the **<MC_AUTDATA_TYP>.<Parameter>** data structure.

Parameter	Description
StartPosition	The StartPosition, which allows changing from basis state 0 to another state at the moment a specific master position is reached. To do this, a corresponding change event with the event type ncSTART must be defined for basis state 0. Specification of the next parameter, StartInterval, is also important.
StartInterval	If the master position has already passed the StartPosition, then the change event ncSTART is generated at the next multiple of the StartInterval.

Table: Overview of basic parameters

Parameter	Description
MaxMasterVelocity	The slave uses the maximum master speed to calculate its compensation gear and to check if its limits have been exceeded. (ACOPOS warning.) This parameter is only required when the compensation gear is being used.
StartState	StartState enables the automat to be started in any state. The automat starts in the basis state 0 if this parameter is not specified.
StartMaRelPos	StartMaRelPos can be used to start in the initial state within the cam profile. StartMaRelPos specifies the master distance relative to the beginning of the cam profile. If a compensation gear is used, it is ignored in the initial state.

Table: Overview of optional parameters

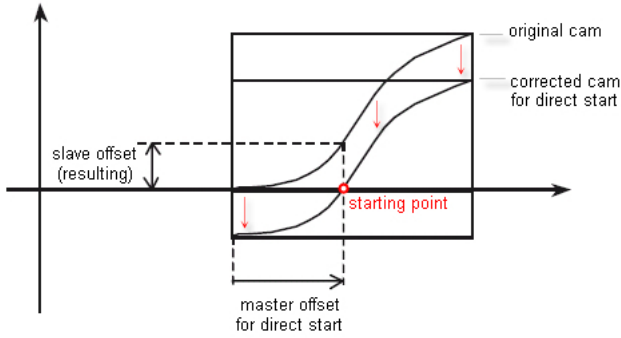
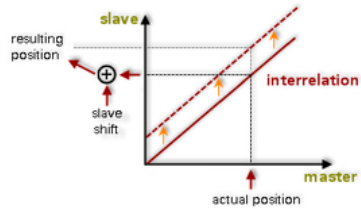

Parameter	Description
	 <p>Representation of a direct start</p>
MasterParID	A ParID can be used as master signal instead of the master axis set position.
AddMasterParID	The value of this ParID is added to the master position.
AddSlaveParID	The value of this ParID is added to the slave position calculated by the automat.
	 <p>Master – Link ratio – Additive element - Slave</p>
SlaveFactorParID	The slave axis scaling is stretched by the value of this ParID. This factor applies to all states in the automat.
EventParID	ParID specification, which serves as event source in states where the event type ncPAR_ID is used. An event is detected if the value of this ParID changes from 0 to a value != 0.
SlaveLatchParID	The slave compensation path begins at the latched value of this ParID in the compensation mode ncSL_LATCHPOS. The value of the ParID specified here (INT type) is latched when a trigger occurs (TRIGGER1, TRIGGER2).

Table: Overview of optional parameters

 [Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ General \ Cam Profile Automat structure](#)

[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ MC_BR_InitAutPar](#)

[Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ Direct start](#)

5.3.2 Cam profile download

A cam profile must first be transferred to the drive via the **MC_BR_DownloadCamProfObj** function block before it can be used in an automat state.

Input	Description
DataObjectName	Name of the cam profile
Index	The cam profile is stored using the index specified on the drive. The cam profile for the corresponding automat state can then be selected using this index.
Periodic	The Periodic parameter can be used to determine whether the cam profile should be executed one time or cyclically. Specification of this parameter is only useful used in combination with the FB MC_CamIn. How a cam profile is processed in the Cam Profile Automat is determined solely by the change event. <ul style="list-style-type: none"> • mcNON_PERIODIC • mcPERIODIC

Table: Overview of the input parameters



A cam profile is selected for an automat state when the states are defined. This cannot be done until a cam profile with the corresponding index is available on the ACOPOS.



Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ MC_BR_DownloadCamProfileData

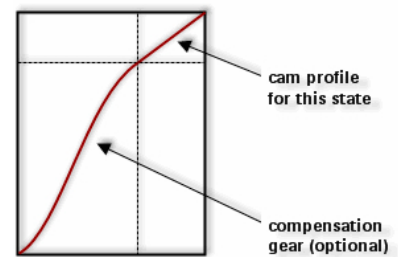
5.3.3 Defining the states

Up to 15 states can be defined. One of the 15 states, the basis state (state 0), is a special case. It is not possible to assign a cam profile or a compensation gear to it. Only the desired change events have to be defined for the basis state. It serves more or less as a waiting step.

The following elements can be defined for the other automat states:

- A cam profile, which must first be transferred to the drive before it can be used. The cam profile can then be used in any state.
- Optionally, a compensation gear that corresponds to an automatically calculated curve and that provides compensation for position and speed differences when transitioning between states. It ensures a continuous link to the cam profile. There are different variations of this ([Linking functions](#))

Automat State



State with compensation and cam profile



It is possible to deactivate the compensation gear. If this is done, then the state only contains the cam profile.

If a compensation gear is used in an automat state, then it will always be processed before the corresponding cam profile in the state.

The **MC_BR_InitAutState** function block and **<MC_AUTDATA_TYP>.State[x]** data structure are used for configuring automat states.

Parameter	Description
StateIndex	The state being handled is specified by StateIndex (1...14).
CamProfileIndex	The CamProfileIndex input is used to select the cam profile for the state.
MasterFactor SlaveFactor	MasterFactor and SlaveFactor define the master and slave-side cam profile scaling.

Table: Overview of basic parameters without compensation

Parameter	Description
RepeatCounterInit	RepeatCounterInit is the initial value for the counter when using the ncCOUNT event type. The counter state is decremented by one each time the end of the state has been reached. The event is generated when the counter reaches zero.
RepeatCounterSet	RepeatCounterSet can be used to change the current counter state on a running automat.

Table: Optional parameters when using the ncCOUNT event



Optional parameters for the use of compensation gears are described in the reference manual in the Automation Studio Online Help documentation.



[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ General \ Automat structure](#)

[Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ Compensation gears](#)

5.3.4 Defining change events

A change event must be defined for a state to induce a state change. Up to 5 change events (0...4) are available for each state.

A change event has the following properties:

- Target state (NextState)
- Event type (Type)
- Event attribute (Attribute)



[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ General \ Automat structure](#)

Target state (NextState)

The target state determines what state should be activated next. The current state can also be selected here for repetition.

Event type (Type)

The event type determines which event triggers a state change. This can be an "external" signal trigger or the end of the current cam profile. A complete list of event types and examples can be found in the Automation Studio Online Help documentation.



[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ General \ Event types](#)

[Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ States \ Events](#)

Event attribute (Attribute)

The event attribute defines the time at which the state change occurs, which is triggered by the corresponding event (action point). This means that the actual state change can be placed at the end of the cam profile when using a trigger as change event, which occurs according to circumstances in the cam profile characteristic.

Event attribute	Description
ncAT_ONCE	The change into the next state is executed immediately or at the beginning of the next sampling cycle.
ncST_END	The change into the next state is not executed before the end of the current state after the compensation and the curve.

Table: Overview of the defined event attributes

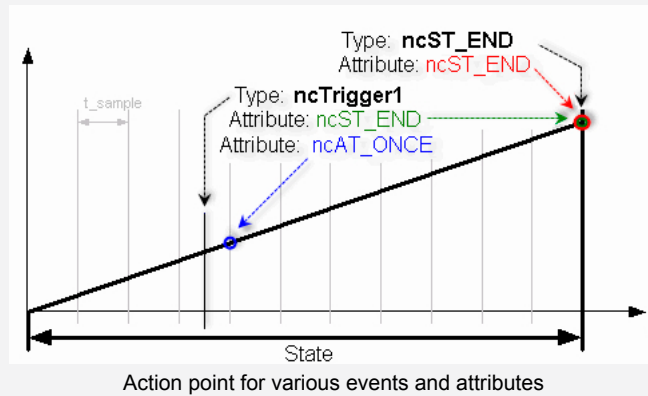


[Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ General \ Event attributes](#)

[Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ States \ Event attributes](#)



The image illustrates how an event attribute works in relation to a change event. A linear curve characteristic is shown which traverses from left to right.



Let's assume that a previously defined change event "Trigger1" occurs within this state. The event attribute ncAT_ONCE is used to immediately change to a defined new state, taking the sampling cycle into consideration.

As a result, the system places the subsequent curve profile precisely at the position of the actual trigger event. Therefore, inaccuracies do not occur in the positioning sequence due to the sampling cycle. When the ncST_END (state end) event attribute is used, this change is made at the end of the current cam profile.

The **MC_BR_InitAutEvent** function block and **<MC_AUTDATA_TYP>.State[x].Event[y]** structure component are used to determine the change event for the automat states and the sequence of the states.

5 change events can be defined for each of the 15 automat states (index 0 to 14). The corresponding indexes are specified on the function block.

Parameter	Description
StateIndex	Specifies which state the event corresponds to.
EventIndex	EventIndex specifies the index for this event.
Type	Type specifies which event type to react to.
Attribute	Attribute determines at what point in time the event should become active. ("action point")
Action	If the Action parameter is set to 1, then this event is also used for synchronized transfer of changed parameters in the automat. See the input parameter Par-Lock from the function block MC_BR_AutControl.
NextState	The parameter NextState specifies which state to change to when the event occurs (target state).

Table: Overview of basic parameters

5.3.5 Starting and controlling the Cam Profile Automat

The function block **MC_BR_AutControl** is used to start and control the Cam Profile Automat. If the automat is configured with the application structure, then this function block also handles initialization of the automat.

Functions that can be activated using MC_BR_AutControl commands:

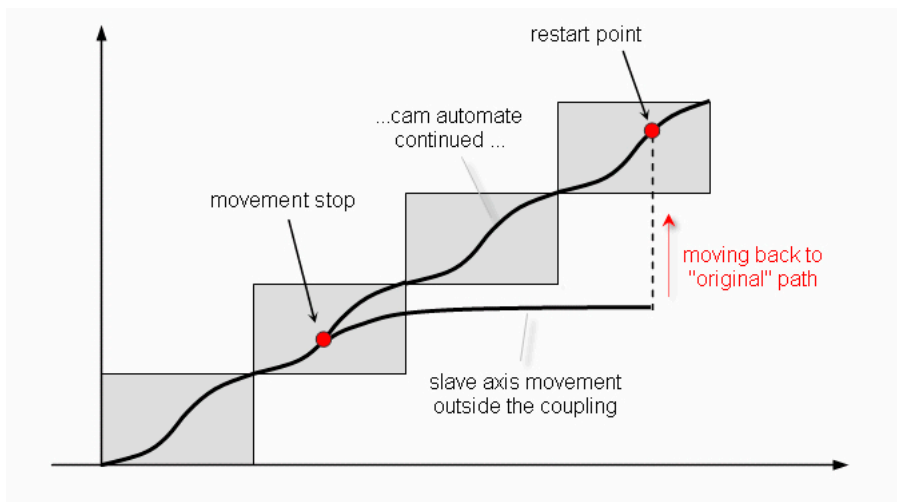
- Online parameter change
- Triggering change events
- Starting and stopping the Cam Profile Automat
- Restarting the automat after a slave "drops out"



MC_BR_AutControl

As an alternative, the following function blocks can be used:

- MC_BR_AutCommand
- MC_BR_InitAutData



Restarting after slave axis failure



Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ MC_BR_AutControl

If the automat is configured with the MC_AUTDATA_TYP data structure, then the Cam Profile Automat parameters defined in the application structure are initialized as follows using MC_BR_AutControl:

Input	Description
AdrAutData	The address of the application data structure is attached to AdrAutData.
InitAutData	The parameter initialization is started with the control command InitAutData.



Online parameter change: The Cam Profile Automat parameters can be changed during run-time. Exceptions to this rule are compensation mode (CompMode), event type (Type), event attribute (Attribute) and MasterParID.

Other methods of stopping the automat: Automat operation can be ended at any time by stopping the slave axis (MC_Stop). Changing the state index to 255 can be used to exit the automat.



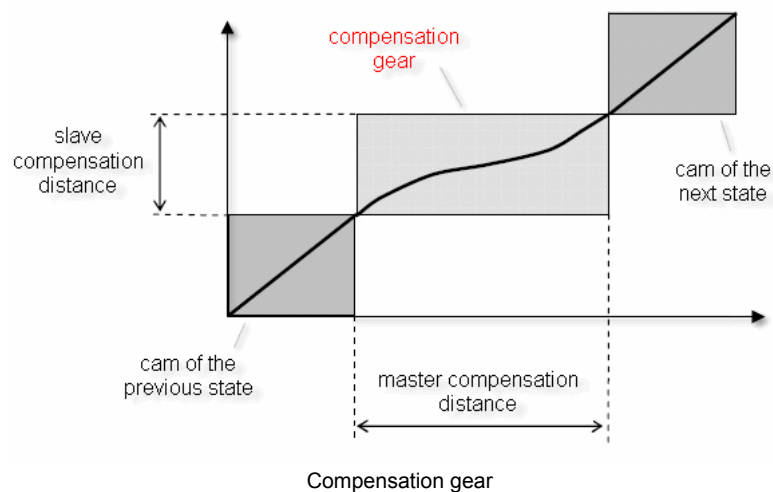
Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ Cam profiles \ Event attributes \ Predefined curves

Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ Online change

5.4 Compensation gear

A compensation gear can be used for each state in the Cam Profile Automat.

The compensation gear is an automatically calculated curve which compensates for position differences during a state transition and ensures smooth transitions between cam profiles. The necessary parameters are provided in the **MC_BR_InitAutState** function block and in the **<MC_AUTDATA_TYP>.State[x]** data structure. (see 5.3.3 "Defining the states")



The image shows compensation between two consecutive states (cam profiles). If compensation is used in a state, then the compensation movement is always performed before the cam profile of the state.

These are the base parameters that define the compensation:

- Compensation mode (CompMode)
- Master compensation distance (MasterCompDistance)
- Slave compensation distance (SlaveCompDistance)

The different compensation gear modes provide possibilities for compensating path as well as speed differences. A precise overview and description can be found in the Automation Studio Online Help documentation.



Programming \ Libraries \ Motion libraries \ ACP_10_MC \ Function blocks \ Cam Automat \ General \ Compensation gears

Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ Compensation gears \ Methods for compensation of position differences

Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ Compensation gears \ Methods for compensation of speed differences

5.5 Examples

Exercise: Using a Cam Profile Automat

Use the Cam Profile Automat with the help of the sample program.

The example and the automat configurations for the flying saw and the cross cutter should be put into operation here.

- 1) Use the single axis example for the master axis
- 2) Import the Cam Profile Automat example for the slave axis
- 3) Select the automat configuration
The automat configuration can be transferred using a function call in the init SP of the sample program.
- 4) Set the axis periods on the master and slave according to the Help documentation
- 5) Analyze the states and events for the selected automat configuration
Additional signals are needed to start a sequence and change the cam profile according to the automat configuration. (MC_BR_AutControl)
- 6) Starting and homing of the master and the slave axis
- 7) Start a positive movement on the master
- 8) Start the Cam Profile Automat on the slave
- 9) Record the master and slave position
- 10) Analyze the results



Programming \ Libraries \ Motion \ Motion control \ Cam Automat \ Automat configuration \

- Labeling machine
- Flying saw
- Cutting unit

Motion \ Reference manual \ ACOPOS drive functions \ Cam Profile Automat \ Examples

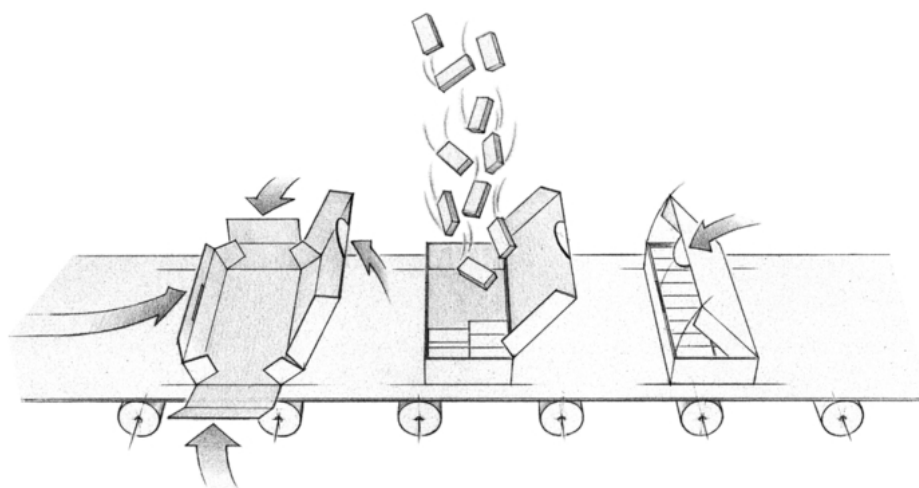
6 SUMMARY

The ACP10_MC library provides numerous functions for linking axis objects. The individual function blocks are designed based on the PLCopen Motion Control standard and feature a uniform design regarding functional usage.

The electronic gear makes it possible to implement linear position links, even with a defined starting point for the axes, if necessary.

Corresponding function blocks are also provided for non-linear position links using electronic cam profiles. The application program controls interactions between multiple cam profiles.

Cam profiles can be created using the cam profile editor in Automation Studio. A wide variety of settings makes it easy to tailor a cam profile to the demands of a particular process. Preset cam profiles on the drive expand the range of functions.



Cartoning

The Cam Profile Automat is an extremely powerful tool for effectively linking cam profiles. The necessary sequences are completely predefined. Initialization of the automat structure and control of the automat mode can be handled using clear and organized functions. Once the Cam Profile Automat is started, the defined sequences are independently processed on the drive. This reduces the load on the application program and results in a very fast, event-controlled positioning sequence.

The ACP10_MC multi-axis functions are subject to the effects of the states in the Motion Control state diagram. The user is provided with necessary information for planning the sequence here.

The ACP10_MC sample programs contain suggestions for specific linking applications and use of the Cam Profile Automat and can be used as templates for creating a complete positioning application.

TRAINING MODULES

TM210 – Working with Automation Studio
TM213 – Automation Runtime
TM220 – The Service Technician on the Job
TM223 – Automation Studio Diagnostics
TM230 – Structured Software Development
TM240 – Ladder Diagram (LD)
TM241 – Function Block Diagram (FBD)
TM242 – Sequential Function Chart (SFC)
TM246 – Structured Text (ST)
TM250 – Memory Management and Data Storage
TM261 – Closed Loop Control with LOOPCONR
TM400 – Introduction to Motion Control
TM410 – Working with Integrated Motion Control
TM440 – Motion Control: Basic Functions
TM441 – Motion Control: Multi-axis Functions
TM450 – ACOPOS Control Concept and Adjustment
TM460 – Initial Commissioning of Motors
TM480 – The Basics of Hydraulics
TM481 – Valve-based Hydraulic Drives
TM482 – Hydraulic Servo Pump Drives
TM500 – Introduction to Integrated Safety
TM510 – Working with SafeDESIGNER
TM530 – Developing Safety Applications
TM540 – Integrated Safe Motion Control
TM600 – Introduction to Visualization
TM610 – Working with Integrated Visualization
TM630 – Visualization Programming Guide
TM640 – Alarms, Trends and Diagnostics
TM670 – Advanced Visual Components
TM810 – APROL Setup, Configuration and Recovery
TM811 – APROL Runtime System
TM812 – APROL Operator Management
TM813 – APROL XML Queries and Audit Trail
TM830 – APROL Project Engineering
TM890 – The Basics of LINUX

